

# A SELF-TEST AND SELF-DIAGNOSIS ARCHITECTURE FOR BOARDS USING BOUNDARY SCANS

Laung-Terng Wang, Michael Marhoefer\*, and Edward J. McCluskey

CENTER FOR RELIABLE COMPUTING, ERL 460

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science  
Stanford University, Stanford, CA 94305-4055, USA

## ABSTRACT

Previous methods proposed for board self-diagnosis do not address interconnect test or separate interconnect test from chip test. This paper presents a low-cost self-test and self-diagnosis architecture for locating both defective chips and bad interconnects on a printed-circuit board. It is assumed that the boundary scan method developed by the Joint Task Action Group (JTAG) is applied to all chips on the board. To achieve high fault coverage, the proposed method uses pseudorandom patterns from a cellular automaton to locate defective chips, and walking sequences to locate bad interconnects.

**Index Terms** — Board self-test, Board self-diagnosis, Interconnect test, Cellular automata.

## I. INTRODUCTION

A board can contain a mix of microprocessors, ASIC chips, memory devices, I/O controllers, surface-mounted technology (SMT) devices, and off-the-shelf SSI/MSI components. All these chips can be designed with or without built-in test features [McCluskey 86]. Although schemes may be available to test each chip, such schemes may not be very useful for diagnosis of boards or systems.

One approach to board-level diagnosis is to use on-board built-in logic block observers (BILBOs) [Kanopoulos 87] for fault isolation to the logic block level. The on-board BILBOs are inserted between logic blocks (ambiguity groups) to which faults are to be isolated. Therefore, board diagnosis down to ambiguity groups is possible. However, this method does not separate faults caused either by bad interconnects or by defective chips, and the required hardware overhead is high, especially, when fault isolation down to the component level is required.

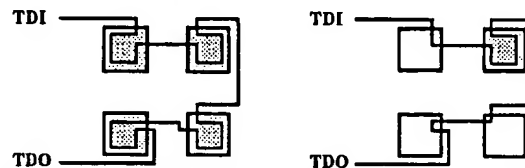
The board self-test architectures proposed in STUMPS [Bardell 87] and MMC [Breuer 88] can be used to isolate faults down to the component level, while at the same time retaining low hardware overhead. STUMPS requires every bistable in a chip be scan-testable. A test chip is produced that contains a parallel test pattern generator (TPG) for test application and a parallel signature analyzer (PSA, aka MISR) for output compaction. MMC requires every chip be testable. A test chip for (module) test and maintenance is also proposed. In both methods, as each chip is testable, diagnosis of defective chips on a board is simple. However, these methods do not address interconnect test and there is a considerable amount of global wiring required to connect the test chip with each of the logic chips.

\* M. Marhoefer was a Visiting scholar at the Center for Reliable Computing (CRC), Computer Systems Laboratory, Stanford University, between 1987 and 1988. He is now with Siemens AG, ZFE F2 DES 233, Otto-Hahn-Ring 6, D-8000 Muenchen 83, FRG.

Turino has proposed the *testability bus* to control and observe many on-board test points via two or more additional chips used for addressing the test points and serial/parallel conversion of test data (see [Turino 84]). The testability bus specification was submitted to IEEE for a standard review [STBS 88]. The hardware overhead associated with this proposal is proportional to the number of test points supported. The degree to which this scheme improves testability at the board and system levels is entirely dependent upon the locations of the selected test points. In most cases, the testability bus can only support tests using an external tester. Self-test or self-diagnosis of a board is very difficult because the chips remain connected with other chips on the board and only a fraction of the chip's pins are typically connected to the testability bus.

A systematic approach to reduce the additional hardware cost is to use *boundary scan*, a test technique submitted to IEEE for a standard review by the Joint Test Action Group (JTAG) [JTAG 88]. This technique has been endorsed by many leading U.S. and European companies. Boundary scan, as its name implies, is a test technique that connects all I/O pads in a scan chain during testing (see Fig. 1a). It does not imply that all internal bistables in the chip must be scan-testable. However, similar to STUMPS and MMC, it allows access to each individual chip. In addition, the JTAG boundary scan technique has a very attractive feature: the ability to bypass I/O pads during board diagnosis (see Fig. 1b).

A major benefit of using this approach is its applicability for many design levels — chip, board and system. Not only does the approach provide an *internal test* of each single chip, but also it provides an *external test* covering the exterior regions between I/O pads on the board. A *sample test* is also possible that allows engineers to take a snapshot of the circuit at a particular instant of time.



(a) A complete serial scan (b) Access to a chip's scan path  
Figure 1. Boundary scan configurations during testing.

The JTAG boundary-scan architecture for one chip is shown in Fig. 2, [JTAG 88]. Four additional pins (connectors) are used to configure the chips for all JTAG-defined and user-defined test modes: Test Data Input (TDI), Test Data Output (TDO), Test Mode Select (TMS), and Test Clock (TCK). The TDI and TDO pins allow the chip to receive test data from and send data to another chip, respectively. The TMS and TCK pins define the test mode to be performed. These four pins constitute a Test Access Port (TAP). The Instruction

Register (IR) stores the test mode to be performed and selects the required test data registers (Boundary Scan Register, Bypass Register, or user-defined registers). The Boundary Scan Register surrounding the functional circuitry of the chip allows for both internal and external tests. The 1-bit Bypass Register is selected during the test of other chips in order to shorten the overall scan path length. Each user-defined register allows the chip to operate in a user-defined test mode, such as self-testing.

The JTAG boundary scan proposal gives specifications at the chip level. No specific information is given as to how this approach is used for board or system test, particularly fault diagnosis. In order to fully utilize the benefits of the JTAG proposal, it becomes imperative to implement an on-board self-test and diagnosis architecture at low cost.

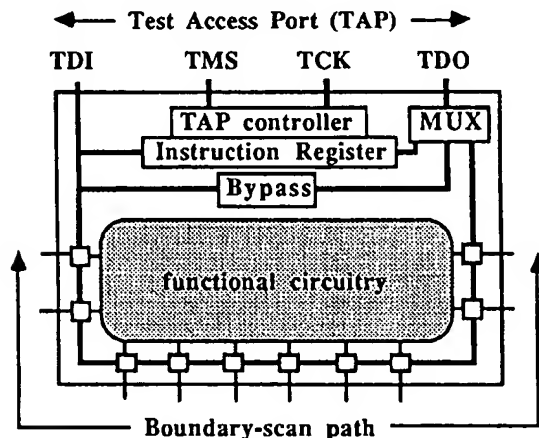


Figure 2. The JTAG boundary-scan architecture for one chip.

This paper will present such an architecture. It is assumed that the board under test is digital and each chip on the board has been designed according to the JTAG boundary scan method. If not, then techniques described in [Beenker 85] and [JTAG 88] can be used to make the chips boundary-scannable. The architecture given below is thus also applicable for testing and diagnosing such chips.

## II. SELF-TEST AND DIAGNOSIS ARCHITECTURE

A low-cost board self-test and diagnosis architecture is presented. This new method guarantees detection and location of defective chips and bad interconnects on a board. The basic idea is to produce a standard test chip that utilizes boundary scan for self-test and self-diagnosis of up to 256 chips on a board.

### 2.1 Proposed Method

Figure 3 shows the test method to self-test and diagnose a board. For simplicity, the method is described for a board with only four chips. In general, the chips on the board are organized logically as a two-dimensional array. The method uses separate scan chains for each column of the array. The input multiplexer transmits test patterns either from the cellular-automaton based (CA-based) pseudorandom TPG or from the walking-0/walking-1 counter to the TDI inputs of the chips in the top row of the array. The output multiplexer, placed in front of the parallel signature

analyzer (PSA), transfers data from the TDO outputs of the chips in the bottom row to the parallel signature analyzer.

The proposed architecture supports three test modes:

- (1) *Board self-test*: All chips belonging to the same row are tested simultaneously. Input patterns are applied through the CA-based TPG to the TDI inputs of the chips in the top row, and output responses are taken from the TDO outputs of the chips for compaction in the PSA.
- (2) *Defective-chip test* (one test mode in board diagnosis): Each chip is tested individually. Input patterns are applied from the CA-based TPG to the TDI input of the column in which the tested chip is located, and output responses are taken from the corresponding TDO output of the column to the PSA. In this case, the other chips are bypassed and the PSA is converted into a serial signature analyzer (SSA).
- (3) *Interconnect test* (another test mode in board diagnosis): Each interconnect is tested individually. Input patterns are applied to the TDI inputs of the chips in the top row from the walking counter, and output responses are taken from the TDO outputs of the chips in the bottom row and stored in the RAM.

In each test, appropriate Boundary Scan Registers (each surrounding the functional circuitry of a chip) and Bypass Registers must be selected. The edge connectors between boards are disconnected so that each board can be tested in isolation. Connection faults between boards (which represent another level of hierarchy) are not considered in this paper. More details of the CA-based TPG and walking-0/walking-1 counter will be presented in Section 3.

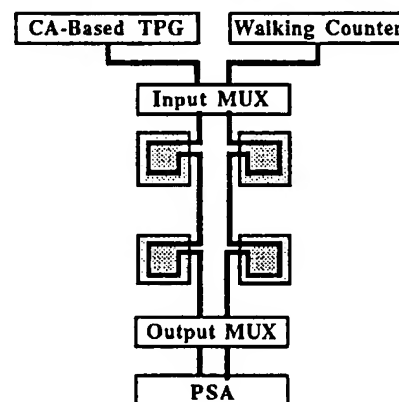


Figure 3. A new architecture to self-test and diagnose chips on an example board containing a 2x2 array of chips.

The proposed architecture is largely based on the following cost observations:

- (1) In production testing, a fast, inexpensive, but comprehensive, *board self-test* must be provided.
- (2) In field service, only fault detection is necessary to reduce system diagnosis time. Defective boards are simply identified, removed and replaced, on-site, with new ones. Fault detection is done using the board self-test method provided for production testing.

Table 1. Board Diagnosis Architectures for Diagnosing $n$ Chips.					
	Method 1	Method 2*	Method 3	Method 4	Method 5
Number of input test pattern generators	one serial	one parallel	$\sqrt{n}$ serial	one parallel	$\sqrt{n}$ serial
Number of output signature analyzers	one serial	one parallel	one parallel	$\sqrt{n}$ serial	$\sqrt{n}$ serial
Number of scan paths (scan chains)	1	$\sqrt{n}$	$\sqrt{n}$	$\sqrt{n}$	$\sqrt{n}$
Number of passes (single-chip failures)	$\log_2 n$	$\log_2 n$	$\log_2 n$	$0.5 \cdot \log_2 n$	$0.5 \cdot \log_2 n$
Number of passes (multiple-chip failures)	$n$	$n$	$n$	$\sqrt{n}$	$\sqrt{n}$
Additional pad (or connector) count	4	$2+2\sqrt{n}$	$2+2\sqrt{n}$	$2+2\sqrt{n}$	$2+2\sqrt{n}$
Scan length for each pass	$O(n)$	$O(\log_2 n)$	$O(\log_2 n)$	$O(\log_2 n)$	$O(\log_2 n)$

\* Our method

- (3) In board diagnosis, fault location (a detailed diagnosis) of defective chips and bad interconnects is only performed off-site at the service or maintenance center. As GO/No-Go test is executed more often than diagnosis, the diagnosis process can be slower in order to reduce the required additional hardware and field maintenance cost.

The proposed method uses a parallel test generator, parallel scan chains, and a parallel signature analyzer for board self-test and self-diagnosis. Depending on need, other configurations are possible. Table 1 compares the requirements of five configurations for a diagnosis of  $n$  chips. This table shows the required hardware overhead and the number of passes (tests of one scan path configuration) necessary to identify the defective chip(s). Under the assumption that only one chip can be faulty, binary search is used to recursively reduce the number of suspects by 50%. The resulting  $\log_2 n$  expression is reduced by an additional 50% for Methods 4 and 5 because all columns (each containing only  $\sqrt{n}$  chips) are individually diagnosed in parallel. Under the assumption of multiple faulty chips, all  $n$  chips in the array (all  $\sqrt{n}$  chips for each parallel, diagnosed column) must be tested. The additional pad count (connector count) assumes the use of the JTAG boundary scan technique and, for the parallel diagnosis of columns, two common signals (TCK, TMS).

In summary, Methods 1 and 2 are the most cost-effective for diagnosis of a single faulty chip in terms of hardware overhead. Although Method 2 yields lower hardware overhead and shorter diagnosis time than Method 1, it requires more pins. For diagnosis of multiple faulty chips, Methods 4 and 5 are much faster but have higher hardware overhead than Methods 1 and 2. The use of separate TPGs in Methods 3 and 5 generate patterns that look "more" random at the circuit inputs because the data dependency which exists between the TPG-generated sequences is removed.

Many benefits are gained when using the proposed method (Method 2) for fault detection and diagnosis:

- (1) It allows fast fault detection in the system.
- (2) It allows diagnosis of multiple chip and interconnect failures on the faulty board.
- (3) The coverage loss problem due to error-masking in the PSA is eliminated since the PSA is actually transformed into a serial signature analyzer during diagnosis.
- (4) Compared to Method 1, its diagnosis time is much shorter.
- (5) Compared to Methods 3, 4 and 5, its hardware overhead is lower.

## 2.2 Architecture

The board under test is organized as a 16-by-16 array containing at most 256 chips. Other array sizes are also possible. In every column of at most 16 chips, the TDO pin of the chip in the  $i^{\text{th}}$  row is connected to the TDI pin of another chip in the  $(i+1)^{\text{th}}$  row, ( $1 \leq i \leq 15$ ), forming 16 distinct scan paths. Test patterns are transmitted to the 16 TDI pins of the chips in the top (first) row. The output responses are scanned out through the 16 TDO pins of the chips in the bottom (16<sup>th</sup>) row. Figure 4 shows the architecture of the Test and Diagnosis Controller (TDC) from a user's point of view. An example implementation of the TDC consists of four parts:

- (1) A standard test chip managing the test data flow and controlling the diagnosis,
- (2) An EEPROM (or EPROM) storing the signatures of the fault-free chips together with the parameters describing the actual dimensions of the 16 boundary scan chains and the on-chip control registers,
- (3) A RAM storing the test and diagnosis results. These results can be read out upon test completion, and
- (4) A set of testability bus (TB) chips that disconnect edge connectors between boards. This allows all boards to be tested simultaneously. Proper signal values must be set during each test. The TB chips can be implemented using the testability bus proposal given in [STBS 88] or simply using the JTAG proposal of [JTAG 88].

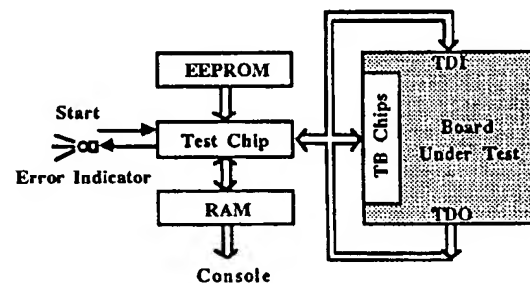


Figure 4. A test and diagnosis controller (TDC).

The EEPROM and RAM can be integrated within the standard test chip depending on the need and cost allowed. The proposed Test and Diagnosis Controller (TDC) can reside on the diagnosed board itself or on an external board. The on-board TDC makes parallel board diagnosis possible,

but has higher hardware overhead since an off-board TDC could be shared by several boards or even systems.

The standard test chip's interface to the board under test has a 16-bit parallel input bus and a 16-bit parallel output bus for boundary scan data. It also has two additional pins, TMS and TCK, for controlling the 16 boundary scan paths. A user views the standard test chip as a set of registers containing one 28-bit Pseudorandom Test Pattern Generator, one 16-bit Input Multiplexer, one 16-bit Output Multiplexer, one 28-bit Parallel Signature Analyzer, one 28-bit Comparator, RAM/ROM Access Logic, a 1-bit Error Indicator, and certain Control Logic to start and stop each test, (see Fig. 5).

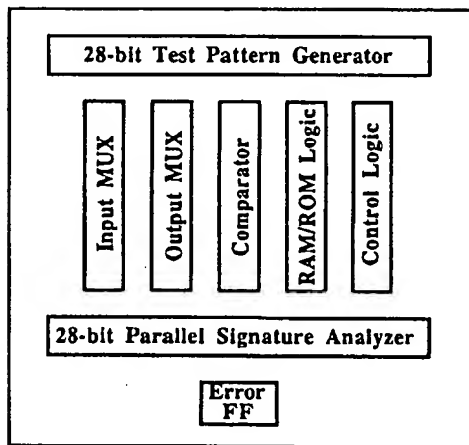


Figure 5. User's architecture for the standard test chip.

During *defective-chip test*, pseudorandom patterns are derived from the rightmost 16 bits of the 28-bit TPG and shifted to the 16 TDI inputs of the chips in the top row (one per column) through the input multiplexer. The output multiplexer selects data from the 16 parallel TDO outputs of the chips in the bottom row (all 16 TDO outputs during board self-test or one TDO output during defective-chip test) and passes them to the parallel signature analyzer (PSA) for output compaction. When the test ends, the computed, final signature is compared with the precomputed, error-free signature stored in the EEPROM. The RAM serves as a bad-chip marker for all of the chips being tested. The error indicator is turned ON whenever there is a discrepancy between the collected signature and its corresponding error-free signature or when the *initial self-test* fails. The RAM/ROM access logic provides data transfers between the TDC and the memories.

The JTAG boundary scan method does not consider how the internal logic of a chip is designed. In case the chip is designed with internal scan (all bistables within the chip are scan-testable) [Bardell 87], each internal scan register can be connected with the Boundary Scan Register to form a serial scan chain. In case of a self-test chip [Eichelberger 83] [LeBlanc 84] [Lake 86], only the user-defined test mode (self-testing) needs to be transferred to the tested chip. There is no need to apply patterns externally. When the test ends, the chip's error signal is simply shifted out through boundary scan. Thus, the proposed architecture is applicable to self-test chips and chips designed with (complete) internal scan.

The pseudorandom test pattern generator (TPG) is constructed using a cellular automaton [Hortensius 87] [Gloster 88], rather than using the conventional LFSR configuration [McAnney 87] [Wang 88]. A cellular automaton based (or CA-based) TPG is chosen here because it (1) provides patterns that look "more" random at the circuit inputs, (2) has higher opportunity to reach full fault coverage in a circuit that is random pattern resistant, and, most importantly, (3) has implementation advantages since it only requires adjacent neighbor communication (no global feedback) and it is *cascadable* -- the physical length of the generator can be increased or decreased by simply adding or deleting cells.

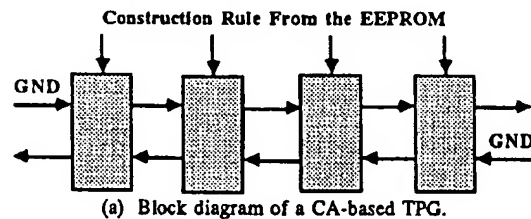
A cellular automaton evolves in discrete steps with the next value of one cell dependent on the previous values of the cell and its left and right cells. It has been shown in [Hortensius 87] that by combining cellular automata rules 90 and 150, given below, one can generate a maximum-length sequence of  $2^s - 1$ , where  $s$  is the number of cells or stages in the test pattern generator.

$$\text{Rule 90: } a_i(t+1) = a_{i-1}(t) + a_{i+1}(t) \quad (1)$$

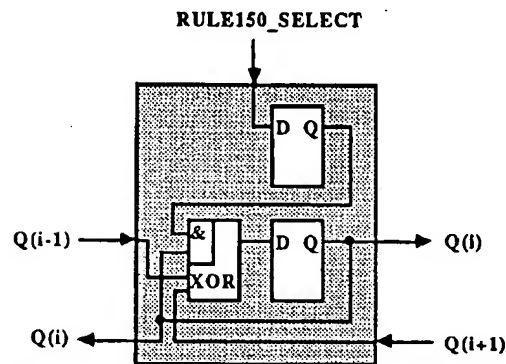
$$\text{Rule 150: } a_i(t+1) = a_{i-1}(t) + a_i(t) + a_{i+1}(t) \quad (2)$$

In Eqs. 1 and 2, the "+" sign denotes modulo-2 addition, and  $a_i(t)$  represents the value of cell  $a_i$ ,  $1 \leq i \leq s$ , at time  $t$ . The construction rules needed to generate maximum-length sequences for  $4 \leq s \leq 28$  can be found in [Hortensius 87] or [Gloster 88].

The CA-based TPG is programmable to generate test lengths of  $2^s - 1$ , for  $16 \leq s \leq 28$ . Figure 6(a) shows a block diagram of the CA-based TPG with the universal CA-cell for generating patterns based on rule 90 or rule 150 given in 6(b). In Fig. 6(b), if the RULE150\_SELECT signal is set to HIGH, then patterns using rule 150 are generated, otherwise rule-90 patterns are generated.



(a) Block diagram of a CA-based TPG.



(b) A CA-cell generating patterns using rule 90 or 150.

Figure 6. A CA-based test pattern generator (TPG).

During *interconnect test*, the CA-based TPG is reconfigured as a walking-0 and walking-1 counter. Test patterns generated from this counter are shifted through the input multiplexer to the selected column. Deterministic walking sequences (rather than pseudorandom patterns) are chosen here because they are I/O-pad order independent, capable of locating all single and multiple interconnect failures due to stuck-at or wire shorts [Hassan 88]. More detailed discussions are given in Subsection 3.3.2.

### III. SELF-TEST AND DIAGNOSIS PROCEDURE

A board with the proposed test and diagnosis controller (TDC) can self-test and self-diagnose itself autonomously. Activated by a start signal (see Fig. 4), the TDC starts with an initial self-test (see Subsection 3.1). After successful completion of this initial test, self-test and diagnosis of the board follow, as described in Subsections 3.2 and 3.3, respectively. As board self-test and diagnosis are completely autonomous, they can be done in parallel for all boards in a system.

#### 3.1 Initial Self-Test

The overall test circuitry includes the TDC and the test circuitry in each boundary scan chip. *Initial self-test of the TDC* checks whether the test circuitry embedded on the board functions correctly. It includes:

(1) Testing the standard test chip: This can be done by testing the operation of few logic blocks at the same time. Consider the case of testing the 28-bit CA-based TPG, its corresponding data paths at the input and output multiplexers, and the 28-bit PSA (see Fig. 3). A direct path, bypassing all boundary scan chips, is established between the outputs of the input multiplexer and the inputs of the output multiplexer. Configurations 1 and 2 of the construction rules for  $s = 28$  [Gloster 88] are successively applied to the TPG for generating pseudorandom patterns. The pattern length required can be determined by fault simulation. These two configurations are necessary because each cell in the TPG can act as a rule-90 or rule-150 cell. The final signature collected in the PSA for each configuration is compared with the predetermined, error-free signature stored in the EEPROM. A non-zero result at the comparator output will signal the malfunction of the tested logic blocks.

To test the walking counter and its associated logic, 32 test sequences (16 pairs of walking-0 and walking-1 patterns) can be used for the 16 data paths (to the PSA). When a walking-0 (or walking-1) pattern is transmitted to a data path, the remaining 15 data paths will each receive an all-one (or all-zero) pattern from the input multiplexer. To test the 1-bit error flip-flop, since the stuck-at-0 fault cannot be detected by any of the normal inputs to the circuit, a testable design technique given in [McCluskey 88] must be followed.

(2) Testing the EEPROM: This can be done by simply compacting the contents of the EEPROM in the PSA. With a pre-stored signature or checksum in the EEPROM, the final signature of the compaction must be zero.

(3) Testing the RAM: This can be done by applying march [Nair 79] or simple march [Sun 84] patterns to the RAM. These patterns can be easily generated (in the RAM/ROM Access Logic) using the self-test technique given in [Sun 84]. Responses coming out of the RAM are directly compared (without compaction) with the original, supplied test pattern set.

(4) Testing the testability bus (TB) chips: The TB chips used here serve only one purpose: to disconnect edge connectors between boards and set proper signal values at the input edge connectors. Thus, testing can be done by scanning in an alternate zeros and ones pattern to all TB chips on the board. (Notice that connection faults between the output edge connectors of a board to the input edge connectors of another board are not considered in this paper.) This pattern is generated in the Control Logic. After the pattern is applied, data latched in the input edge connectors is scanned out through these TB chips and directly compared with the original test pattern. A non-zero result will indicate a stuck-at error in the TB chips, and the error indicator is turned on.

*Initial self-test of the test circuitry embedded in each boundary scan chip* is performed after the initial self-test of the TDC is successfully completed. It includes:

(1) Testing all Bypass Registers: A deterministic control sequence stored in the EEPROM is shifted via the scan path into the Instruction Registers of all chips. Loading the Instruction Registers is controlled by the signals TMS and TCK. This sequence selects all Bypass Registers. Testing is then done by sending an alternate zeros and ones pattern to the TDI pins of the 16 scan chains. The output results shifted out of the TDO pins are then directly compared with the original data inputs. If a stuck-at fault is detected, the error indicator is turned on.

(2) Testing all Boundary Scan Registers: The test for Bypass Registers is repeated for Boundary Scan Registers except that the control sequence is derived from the boundary scan parameters not from the bypass parameters.

(3) Testing the sample mode: This only requires toggling the output state of each input or output register. Testing can be done using an approach similar to board self-test (given in Subsection 3.2) except that the test can stop when all nodes at the input and output registers have been toggled. This requires using a logic simulator to obtain the desired test length.

(4) Testing all user-defined test modes: This can be done by using the EEPROM to store all required control sequences. As IC manufacturers may not be willing to disclose these modes, it is difficult for a board or system designer to implement tests for them.

#### 3.2 Board Self-test

Upon successful completion of the initial self-test, board self-test is started. Board self-test provides a quick Go/No-Go test for all chips and interconnects on the board. All 16 chips belonging to the same row are tested at the same time. The Boundary Scan Registers are selected for the 16 chips being tested and the Bypass Registers are selected for the remaining chips.

During each test application, a pattern generated in the CA-based TPG is applied to the 16 TDI inputs of the chips in the top row and shifted out through the bypass paths (not boundary scan paths) to the 16 TDI inputs of the chips in the row being tested. The system clock is executed once. The output response is then taken from the 16 TDO outputs of the chips in the bottom row being tested and scanned out through the output multiplexer to the PSA. When the test ends, the final signature collected in the PSA is compared with the good, precomputed signature stored in the EEPROM. The error indicator is turned on whenever a discrepancy is found, and the bad board position is stored in the RAM for future reference.

This approach requires the input edge connectors (connected to the board under test) be disconnected so that all boards can be tested simultaneously. The TB chips of Fig. 4 provides this isolation.

Board self-test time can be estimated by the time for one pass (the test of 16 chips in one row via the scan pass) multiplied by the number of passes, neglecting the time to reconfigure the scan path. If, for example, each chip has an average of 30 pins, then the average scan path in a column will have  $30 + 15 = 45$  bits. This includes the bypass bits. Therefore, the total diagnosis time,  $T$ , for a  $16 \times 16$  array of such chips, applying  $k$  pseudorandom patterns per pass (row) and using a 10 MHz clock signal TCK, can be estimated by the formula:

$$T = 16 \text{ (passes)} \cdot k \text{ (patterns/pass)} \cdot 45 \text{ (bits/pattern)} / 10^7 \text{ (bits/s)}.$$

For  $k = 2^{20} - 1 = 10^6$  pseudorandom patterns, the board self-test time would be close to 75 seconds.

To reduce system down time, these bad boards are immediately removed and replaced. Another run of board self-test is then restarted. This process repeats until the system is up again. Comprehensive bad-board diagnosis can then be performed at the field service center.

### 3.3 Board Diagnosis

After a successful board self-test, board diagnosis can be started either on-site or at the field service center. Diagnosis on a board includes defective-chip test and board interconnect test. The defective-chip test tries to identify those faulty chips, while the interconnect test detects and locates one or more bad interconnects due to stuck-at faults and wire shorts.

#### 3.3.1 Defective-Chip Test

Defective-chip test is performed when a chip is operated in *internal test* mode with the remaining chips in bypass mode. Each chip is tested individually. For this purpose, each of the 256 chip tests starts with selecting, in every column, the Boundary Scan Register for the chip in that row and selecting the Bypass Registers for all other chips. Similar to initial self-test and board self-test, this is also accomplished by shifting a deterministic control sequence, stored in the EEPROM, via the scan path into the Instruction Registers. Depending on the instruction code shifted in, the Boundary Scan Register is selected in one chip, (an *internal test* mode is activated), and the 1-bit Bypass Register in all others. Using such a serial scan test approach, pseudorandom patterns are applied from the TPG to the chip being tested and output responses are shifted out through the output multiplexer to the PSA. The PSA, reconfigured as a serial signature analyzer, computes a signature for every single chip.

At the end of each chip test, the signature is written into the RAM and compared with the signature stored in the EEPROM for the fault-free chip. The result of this comparison is stored in the RAM. After the test of the last chip, the diagnosis of multiple-chip failures is completed. These multiple-chip failures cannot mask each other since each chip is tested individually. In addition, for better diagnosis result, a different test length for each chip is used and stored in the EEPROM.

Diagnosis time can be estimated by the formula:

$$T = 256 \text{ (passes)} \cdot k \text{ (patterns/pass)} \cdot 45 \text{ (bits/pattern)} / 10^7 \text{ (bits/s)}.$$

For  $k = 2^{17} - 1 = 131,000$  pseudorandom patterns per chip, the board diagnosis time would be close to 150 seconds. There may be a problem when more than  $2^{17}$  (or  $10^5$ ) patterns are needed; however, it is unlikely that permanent faults (due to stuck-at or wire shorts) will not manifest themselves during the application of  $10^5$  patterns. If it is important to reduce the chip diagnosis time, then 16 serial signature analyzers (as suggested in Method 4 of Subsection 2.1) instead of one parallel signature analyzer can be used. In this case, the chip diagnosis time will be decreased by 16 times, while the data storage needed for signature comparison is increased by 16 times.

#### 3.3.2 Interconnect Test

Interconnect test identifies one or more bad interconnects when all chips are operated in *external test* mode. Faults considered in this test include all single and multiple stuck-at and 2-wire bridging faults on the board interconnects. For 2-wire bridging faults (wire shorts), both wire-AND shorts (where logic zero dominates) and wire-OR shorts (where logic one dominates) can occur [Millman 88]. Thus, these types of wire shorts are considered.

The complete boundary scan is used for board interconnect test. This is activated by selecting all Boundary Scan Registers and an *external test* mode. Let the number of I/O scan pads on the board be  $N$ . Assume that there are no bidirectional pads, and none of the tri-state buses (connecting outputs of several drivers) are forced to a high impedance state at any instant of time. It has been shown in [Hassan 88] that by connecting all the chips to form a single scan chain,  $2N$  patterns ( $N$  pairs of a walking-0 pattern followed by its corresponding walking-1 pattern) are capable of locating all single and multiple stuck-at faults and wire shorts on the interconnects. A walking-0 (or walking-1) pattern is an all-one (or all-zero) pattern except the  $i^{\text{th}}$  bit,  $1 \leq i \leq N$ , is set to a logic zero (or one) value. The logic zero (or one) value shifted to the  $i^{\text{th}}$  pad will locate the interconnect failure in that pad due to a wire-OR (or wire-AND) short. Thus, this pattern set (see Table 2) is structure-independent. The scheme requires a time complexity of  $O(N^2)$  for shifting the  $2N$  patterns of  $N$  bits each into and out of the board.

Table 2. Patterns to Test Board Interconnects of  $N$  Wires.

No.	Test Patterns	Diagnosed Interconnect Faults
(1)	1000 .... 0000	sa0 & wire-AND shorts in No. 1
(2)	0111 .... 1111	sa1 & wire-OR shorts in No. 1
(3)	0100 .... 0000	sa0 & wire-AND shorts in No. 2
(4)	1011 .... 1111	sa1 & wire-OR shorts in No. 2
:	:	:
(2N-1)	0000 .... 0001	sa0 & wire-AND shorts in No. N
(2N)	1111 .... 1110	sa1 & wire-OR shorts in No. N

In this paper, we propose using multiple scan chains rather than a single scan chain. The time complexity is reduced to  $O(N^2)/p^2$ , when  $p$  scan chains,  $p \geq 1$ , are used. As most boards use bidirectional cells for data transfer, interconnect diagnosis of bidirectional cells is also considered.

Since each bidirectional cell can act as an input cell or an output cell, complete diagnosis for boards containing bidirectional cells will require  $4N$  test patterns (walking sequences). The  $2N$  patterns listed in Table 2 are applied twice: one with each bidirectional cell as an input (or output) cell and the other as an output (or input) cell. The EEPROM contains the required sequences for the reconfiguration.

As the proposed board self-test and diagnosis architecture contains 16 scan chains, it is advantageous to shift patterns in parallel rather than in series. Let  $N'$  be the maximum number of I/O (scan) pads in the scan chain,  $N' \geq N/16$ . Figure 7 shows an example 4-bit walking-0/walking-1 counter reconfigured from the CA-based TPG, when  $N' \leq 2^4 - 1$ . When the WALKING\_ONE control signal is set to one (or zero), the counter generates an  $N'$ -bit walking-1 (or walking-0) sequence and applies them through the input multiplexer to one of the selected scan chain (see Fig. 3). At the same time, the input multiplexer sends an all-one (or all-zero) pattern to the remaining columns.

The response after the application of a walking-0 pattern to the  $N$  pads (in  $N'$  cycles) is shifted out and stored in the RAM, when the next  $N'$ -bit walking-1 pattern is shifted into the chips (see Fig. 8). The stored result is then scanned out in parallel from the RAM and compared (XNORed) with the response of the second pattern. Under the error-free condition, the output response from the first pattern is complement to that from the second pattern. This requires that the TB chips receive an all-one (or all-zero) pattern at the input edge connectors from the Control Logic when a walking-0 (or walking-1) pattern is applied to the  $N$  pads. As a direct comparison method (equality check) is used, there will be no aliasing and error-masking problems due to output compaction [McCluskey 86].

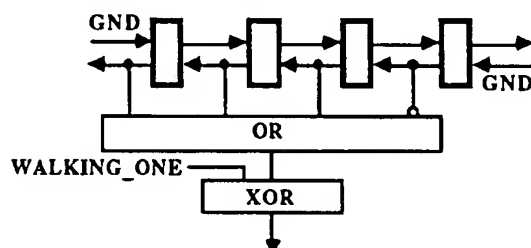


Figure 7. A 4-bit walking-0/walking-1 counter.

Consider the same example given for defective-chip test. Each of the 16 columns has, on average, 16 chips of 30 I/O cells, resulting in a total of  $N' = 480$  scan cells. To generate a walking sequence of 480 bits for that column, it will require a walking-0/walking-1 counter of 9 bits, since  $480 < 2^9 - 1 = 511$ . As  $4N'$  patterns of  $N'$  bits are required, the diagnosis time can be estimated by the formula:

$$T = 4N' (\text{patterns}) \cdot N' (\text{bits/pattern}) / 10^7 (\text{bits/s}).$$

For  $N' = 480$ , the board diagnosis time would be close to 0.1 second. Since the interconnect diagnosis time is much shorter than that taken during board self-test or defective-chip diagnosis, interconnect test is executed before the board self-test and chip diagnosis are performed.

The proposed diagnosis method requires that none of the tri-state buses and bidirectional cells be in a high impedance state. This means that at any instant of time during this test, one and only one tri-state cell in each bus must be enabled, and each bidirectional cell must be configured as an input or output cell. The two predetermined sequences stored in EEPROM ensure the above.

The proposed method only considers stuck-at faults and wire shorts on board interconnects. More detailed treatments in detecting and locating these faults together with other faults, such as stuck-open faults, can also be found in [Wagner 87] and [Hassan 88]. Implementation issues were also discussed in [Hassan 88].

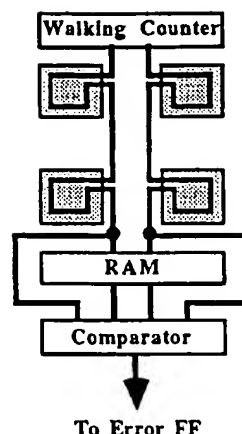


Figure 8. Interconnect diagnosis using an external RAM for storage (on an example board with a 2x2 array of chips).

### 3.4 Control of Parallel Boundary Scans

During defective-chip test, each chip is tested individually. As the chips may have differing numbers of I/O pads, the required number of scan clock cycles for each chip will vary. Performing such a scan test on the board will become a complicated task unless some constraints are imposed on the self-test method.

To simplify the task, the largest number of I/O pads in each row, increased by 15 bits for the Bypass Register, is chosen as the scan test length for each data pattern. This will require scanning additional input data to some chips and collecting additional responses in the signature analyzer. However, by selectively gating the responses to the signature analyzer, one still obtains a final signature that does not include any additional output response. The proposed standard test chip contains a signature control unit (in the Control Logic) that generates the selective gating signals for the signature analyzer. Although this scheme will make the standard test chip design more complex, it provides board designers with a simple scan clock connection mechanism. Hence, the designer can tie all scan clocks (TCKs) together.

### 3.5 Remarks on the Suitability of the JTAG Proposal for Board Diagnosis

The objective of the JTAG proposal is to provide an integrated solution covering tests in all levels, from chip, board to system. In applying this proposal for board self-test and diagnosis, two problems were observed:

First, no design guidelines, such as those of [Nadig 77], were offered for signature analysis or other output compaction schemes. Whenever the board contains tri-state buses in a chip or cannot be initialized to a known state before the test starts, it is likely that any board self-test or diagnosis method which uses an output compaction scheme will fail to function. This limits the JTAG proposal to boards where a time-consuming, direct output comparison technique must be employed. In order to ensure board diagnosability using output compaction schemes and to reduce data storage and diagnosis time, it is suggested that design guidelines be included in the proposal.



Second, no standards for Instruction Register codes have been defined for user-defined test modes. At present, only two instruction codes were reserved: all zeros for selecting the Boundary Scan Register in the external test mode and all ones for selecting the Bypass register. There are no reserved codes for self-test, scan, or internal test modes. This creates some extra overhead during board diagnosis, because all the different instruction codes have to be stored in the EEPROM and have to be looked up when configuring the scan paths. To make board diagnosis more consistent, it is suggested that all of these commonly used test modes be included in the specification.

#### IV. SUMMARY AND CONCLUSION

A low-cost board self-test and self-diagnosis architecture implementing the JTAG boundary scan proposal was presented. Based on this architecture, a standard test chip is proposed to locate defective chips and bad interconnects.

For defective chip diagnosis, pseudorandom test patterns are generated from a cellular automaton in order to provide good fault coverage. The CA-based test generator contains no global feedback, thus allowing maximum-length sequences to be generated for any desired number of cells (flip-flops). This is not the case when LFSR-based test generators are used [Bhavsar 85], where only even (or odd) number of cells can be cascaded to generate maximum-length sequences.

For bad interconnect diagnosis, walking sequences are provided and a direct output data comparison method is used. This method allows all bad interconnects due to wire shorts and multiple stuck-at faults to be located.

The proposed method uses a cellular automaton to generate pseudorandom patterns for board self-test and defective-chip diagnosis. The effectiveness of this method depends on the type of circuits to be tested. For highly regular circuits, such as PLAs, this may not be a viable approach [Eichelberger 80]. However, for random logic circuits, (especially circuits which are random pattern resistant), good fault coverage has been demonstrated [Gloster 88]. It is suggested that for those highly regular circuits and circuits that yield low coverage, a specific self-test method associated with each circuit be employed during the IC design process.

#### ACKNOWLEDGMENTS

The authors wish to thank Dr. R.E. Tulloss of AT&T for sending us the latest JTAG proposal. They also would like to thank Dr. K.D. Wagner, Dr. J.G. Udell, Jr., S.R. Makar, S.D. Millman, L. Avra, I. Shperling, R.C. Anderson, H. Hao, J. Kouloheris, and Dr. B. Marhofer for their valuable comments and suggestions. The help from S.V. Munda and H.-J. C. Wang in preparing this paper is also deeply appreciated. This paper was supported in part by the National Science Foundation under Grant No. MIP-8709128, and in part by the Deutsche Forschungsgemeinschaft, Federal Republic of Germany (FRG).

#### REFERENCES

- [Bardell 87] Bardell, P.H., W.H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*, John-Wiley and Sons, Inc., NY, 1987.
- [Beenker 85] Beenker, F.P.M., "Systematic and Structured Methods for Digital Board Testing," *Proc., IEEE 1985 Int'l Test Conf. (ITC-85)*, pp. 380-385, Philadelphia, PA, Sept. 1985.
- [Bhavsar 85] Bhavsar, D.K., "Concatenable Polydividers: Bit-Sliced LFSR Chips for Board Self-Test," *Proc., ITC-85*, pp. 88-93, Philadelphia, PA, Sept. 1985.
- [Breuer 88] Breuer, M.A., and J.-C. Lien, "A Test and Maintenance Controller for a Module Containing Testable Chips," *Proc., ITC-88*, pp. 502-513, 1988.
- [Eichelberger 80] Eichelberger, E.B., and E. Lindbloom, "A Heuristic Test-Pattern Generator for Programmable Logic Arrays," *IBM J. Res. Develop.*, pp. 15-22, Jan. 1980.
- [Eichelberger 83] Eichelberger, E.B., and E. Lindbloom, "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test," *IBM J. Res. Develop.*, vol. 27, No. 3, pp. 265-272, Mar. 1983.
- [Gloster 88] Gloster, C.S., Jr., and F. Brglez, "Boundary Scan with Cellular-Based Built-In Self-Test," *Proc., ITC-88*, pp. 138-145, Washington, D.C., Sept. 1988.
- [Hassan 88] Hassan, A., J. Rajski, and V.K. Agarwal, "Testing and Diagnosis of Interconnects Using Boundary Scan Architectures," *Proc., IEEE 1988 Int'l Test Conf. (ITC-88)*, pp. 126-137, Washington, D.C., Sept. 1988.
- [Hortensius 87] Hortensius, P.D., "Parallel Computation of Non-Deterministic Algorithms in VLSI," Ph.D. Dissertation, University of Manitoba, Canada, 1987.
- [JTAG 88] Joint Test Action Group (JTAG), "JTAG Boundary-Scan Architecture Standard Proposal," Version 2.0, Mar. 30, 1988.
- [Kanopoulos 87] Kanopoulos, N., N. Vasanthavada, J. Watterson, and J.J. Hallenbeck, "A New Implementation of Signature Analysis for Board Fault Isolation Testing," *Proc., ITC-87*, pp. 730-736, Sept. 1987.
- [Lake 86] Lake, R., "A Fast 20K Gate Array with On-Chip Test System," *VLSI Systems Design*, pp. 46-55, June 1986.
- [LeBlanc 84] LeBlanc, J.J., "LOCST: A Built-In Self-Test Technique," *IEEE Design & Test of Computers*, pp. 45-52, Nov. 1984.
- [McCluskey 86] McCluskey, E.J., *Logic Design Principles: With Emphasis on Testable Semicustom Circuits*, Prentice-Hall, Englewood, NJ, 1986.
- [McCluskey 88] McCluskey, E.J., "Design Techniques for Testable Embedded Error Checkers," Center for Reliable Computing (CRC) Technical Report No. 88-4, Computer Systems Lab., Stanford University, June 1988.
- [Millman 88] Millman, S.D., and E.J. McCluskey, "Detecting Bridging Faults with Stuck-At Test Sets," *Proc., ITC-88*, pp. 773-783, Sept. 1988.
- [Nadig 77] Nadig, H.J., "Signature Analysis - Concepts, Examples, and Guidelines," *Hewlett-Packard Journal*, pp. 15-21, May 1977.
- [Nair 79] Nair, R., "Comments on 'An Optimal Algorithm for Testing Stuck-At Faults in Random Access Memories'," *IEEE Trans. on Computers*, Vol. C-28, No. 3, pp. 258-261, Mar. 1979.
- [STBS 88] "Standard Testability Bus Specification - P1149," Draft 5, May 31, 1988.
- [Sun 84] Sun, Z., and L.-T. Wang, "Self-Testing of Embedded RAMs," *Proc., ITC-84*, pp. 148-156, 1984.
- [Turino 84] Turino, J., "A Totally Universal Reset, Initialization and Nodal Observability Circuit," *Proc., ITC-84*, pp. 878-883, Philadelphia, PA, Oct. 1984.
- [Wagner 87] Wagner, P.T., "Interconnect Testing with Boundary Scan," *Proc., ITC-87*, pp. 52-57, Sept. 1987.
- [Wang 88] Wang, L.-T., and E.J. McCluskey, "Hybrid Designs Generating Maximum-Length Sequences," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 7, No. 1, pp. 91-99, Jan. 1988.